Tokenization Changes Meaning in Large Language Models: Evidence from Chinese

David A. Haslett

Hong Kong University of Science and Technology, Division of Social Science
haslett@ust.hk

Large language models segment many words into multiple tokens, and there is mixed evidence as to whether tokenization affects how state-of-the-art models represent meanings. Chinese characters present an opportunity to investigate this issue: They contain semantic radicals, which often convey useful information; characters with the same semantic radical tend to begin with the same one or two bytes (when using UTF-8 encodings); and tokens are common strings of bytes, so characters with the same radical often begin with the same token. This study asked GPT-4, GPT-4o, and Llama 3 whether characters contain the same semantic radical, elicited semantic similarity ratings, and conducted odd-one-out tasks (i.e., which character is not like the others). In all cases, misalignment between tokens and radicals systematically corrupted representations of Chinese characters. In experiments comparing characters represented by single tokens to multi-token characters, the models were less accurate for single-token characters, which suggests that segmenting words into fewer, longer tokens obscures valuable information in word form and will not resolve the problems introduced by tokenization. In experiments with 12 European languages, misalignment between tokens and suffixes systematically corrupted categorization of words by all three models, which suggests that the tendency to treat malformed tokens like linguistic units is pervasive.

1. Introduction

Large language models (LLMs) process text as a sequence of tokens, and many words are segmented into multiple tokens. For example, GPT-4, GPT-4o, and Llama 3 represent *detokenization* with three tokens: *det* + *oken* + *ization*. (Examples in the Roman alphabet include word-initial whitespace, e.g., *␣det*.) Tokens obscure information in word form, such as the constituents *de* and *token*, but recent work argues that LLMs are robust against misleading tokenization (e.g., Kaplan et al. 2024; Gutiérrez, Sun, and Su 2023). On the other hand, it's unclear how LLMs could learn the meanings of rare and complex words if not from subword constituents. Chinese characters illustrate both the risks and opportunities of inferring meanings from tokens. Chinese characters contain semantic radicals, which often imply semantic features (e.g., the characters for 'tea' and 'stem', 茶 and 茎, contain the 'grass' radical, ⺾), and, for historical reasons, radicals often

Action Editor: Xuanjing Huang. Submission received: 13 November 2024; revised version received: 6 January 2025; accepted for publication: 15 February 2025.

https://doi.org/10.1162/coli_a_00557

correspond to bytes in characters, which sometimes correspond to LLM tokens. Semantic radicals are analogous to suffixes, such as "-ons" in French, which marks verbs in the plural first person, and as with radicals, tokens correspond imperfectly to suffixes. This study investigated whether GPT-4, GPT-4o, and Llama 3 use tokens to represent meanings in Chinese and in 12 European languages.

1.1 Tokenization in Large Language Models

LLMs are often described as learning patterns in words (e.g., Bubeck et al. 2023), and they build on earlier neural networks designed to represent word meanings (e.g., Lenci et al. 2022). However, an LLM's vocabulary comprises tokens, not words, and state-of-the-art LLMs segment most words into multiple tokens, seemingly in all languages (Haslett and Cai 2025; Petrov et al. 2024). For example, GPT-4 represents *annoyingly* and *encodings* as two tokens each: *annoying* + *ly* and *enc* + *odings*. Understanding how LLMs represent multi-token word meanings is necessary to interpret them accurately and to identify when and why representations in LLMs differ from human interpretations.

Presumably, LLMs memorize how to spell some words token by token. For example, GPT-4 might learn that *misgovern* is spelt *mis* + *g* + *overn*. But memorization cannot account for previously unseen words, which abound because language is productive (cf. Baayen 2001), and, even when memorization is feasible, encountering rare words in few contexts will skew their meanings. For humans, not only is memorization unnecessary to interpret *misgovern* (Goulden, Nation, and Read 1990), but opting for memorization over inference would deprive that word of the semantic depth conferred by *govern* and *government* and would amplify idiosyncratic associations with unrepresentative contexts. People therefore decompose rare words and associate them with similar-sounding words (e.g., Alegre and Gordon 1999; Haslett and Cai 2023; for a review, see Amenta and Crepaldi 2012).

Subword tokens are supposed to address the problem of representing rare words (Sennrich, Haddow, and Birch 2016; Wu et al. 2016). One technique for representing word meanings, FastText, segments unfamiliar words into subword strings, which predicts human processing effects (Bojanowski et al. 2017; Gatti, Marelli, and Rinaldi 2023; see also Baayen et al. 2019; Cassani, Chuang, and Baayen 2020; Hendrix and Sun 2021; Marelli, Amenta, and Crepaldi 2015), so subword tokens seem to be on the right track. However, FastText segments words into overlapping constituents (e.g., *encodings* comprises *encod*, *ncodi*, *codin*, etc.), whereas LLMs segment words into non-overlapping, non-decomposable constituents, so the utility of subword tokens depends on the identities of those tokens. In the documentation for two popular tokenization resources, OpenAI and HuggingFace claim that complex words, such as *encoding* and *annoyingly*, should decompose to meaningful constituents, such as *ing* and *ly* (`github.com/openai/tiktoken`; `huggingface.co/transformers`). But in standard tokenization methods such as byte-pair encoding, words are segmented into common strings of bytes, regardless of semantics (for a review, see Mielke et al. 2021), so as *enc* + *odings* and *mis* + *g* + *overn* illustrate, tokens do not reliably correspond to morphemes (e.g., *ex-*, *un-*, and *-able*; Bostrom and Durrett 2020; Church 2020). Increasing an LLM's vocabulary of tokens entrenches the problem of extracting meaning from word form: *government*, *governance*, and *governor* are each a single token to GPT-4, GPT-4o, and Llama 3, so they don't decompose to *govern*, either.

There's some evidence that LLMs represent multi-token words compositionally and benefit from meaningful constituents. Several studies report that tokens that correspond to morphemes improve LLM performance on a variety of tasks (e.g., matching

words with definitions; Batsuren et al. 2024; Jabbar 2023; Mager et al. 2022), but in other cases, they don't confer any advantage, or the results are mixed (Arnett et al. 2024; Gutiérrez, Sun, and Su 2023). Hofmann, Pierrehumbert, and Schütze (2021) found that when morpheme-like tokens do improve performance, LLMs benefit mostly from stems (i.e., root words) rather than affixes (e.g., *pre-* and *-ing*). The presence of root words is a feature of those methods, not a bug, but it weakens evidence that LLMs extract semantic information from subword tokens because, in effect, morphological tokenization turns multi-token words back into their single-token roots. If LLMs do take advantage of morpheme-like tokens to represent meanings, the question remains: How do they represent the many words that comprise linguistically malformed tokens?

Morphemes and meaningless strings of letters lie on a continuum, so non-morphemic tokens can convey valuable information (for reviews, see Dingemanse et al. 2015, and Haslett and Cai 2024). For example, OpenAI implies, in the documentation for the tiktoken Python package, that *enc* + *odings* is an unhelpful tokenization because it doesn't align with morphology, yet from the perspective of GPT-4, it shares a constituent with *enc* + *oders*, *enc* + *od* + *ified*, and *enc* + *od* + *ification*. In this way, across many languages, byte-pair encoding conveys morphological information even when tokens don't align with morphemes, and it captures other sorts of information in word form, such as etymological relationships (Haslett and Cai 2025). So, LLMs should be able to extract semantic information from standard frequency-based tokens, which might explain why morphological tokenization doesn't always improve performance.

Alternatively, LLMs might defy analogies to compositionality in human language processing. A recent line of research emphasizes how the early layers of LLMs abstract meanings away from particular tokens, such that the representations of multi-token words are better understood as distributed throughout an LLM than as a collection of contextualized token embeddings (Elhage et al. 2022; Gurnee et al. 2023). Several studies have argued that this "detokenization" makes LLMs robust against the vagaries of subword tokens. For example, GPT-4 segments *encodings* differently when it's capitalized versus not (i.e., *enc* + *odings* versus *EN* + *COD* + *INGS*), but LLMs seem to understand that such near-duplicate forms have equivalent meanings (Feucht et al. 2024; Kaplan et al. 2024). On the other hand, treating near-duplicates as identical hinders LLM performance (Schäfer et al. 2024), which suggests that superficial differences in tokenization lead to important differences in representations. More to the point, detokenization skirts the question of where word meanings come from. How can an LLM effectively represent *encodification*, for instance, without identifying constituents such as *code*, and how do the tokens *enc* and *od* not distort or obscure that information? These are not rhetorical questions. LLMs are powerful black boxes, and it's entirely possible that they represent complex words in a way that makes subword tokens irrelevant. In fact, LLMs can infer the identities of letters in tokens far better than expected by chance (Edman, Schmid, and Fraser 2024; Itzhak and Levy 2022; Kaushal and Mahowald 2022), and they could use that sub-token information to represent complex or unfamiliar words.

1.2 Semantic Radicals in Chinese Characters

Chinese characters contain constituents called **radicals**. There are 214 radicals, which can imply function (semantic radicals) or pronunciation (phonetic radicals). For example, the 'mouth' radical, 口, is phonetic in 加 ('add') and is semantic in 吗 (a question particle). As shown in Table 1, grammatical particles often contain 'mouth' as a semantic radical, verbs often contain 'hand', and plants often contain 'grass'. Chinese characters are monosyllabic morphemes, so semantic radicals are sub-morphemic constituents,

Table 1
Examples of Chinese characters, semantic radicals, and token IDs. Bytes, in parentheses below token IDs, are hexadecimal digits. A pair of hexadecimal digits represents a single byte (i.e., six hexadecimal digits represent the three bytes in each character). GPT-4o and Llama 3 have larger vocabularies of tokens than GPT-4 does, so here, only the 'stem' character comprises multiple GPT-4o and Llama 3 tokens. Notice that the final byte in 'stem' has the same token ID in all three LLMs (236). This is because their vocabularies begin with the same set of 256 bytes, arranged in the same order, so byte "8e" has the same index in all three token vocabularies. In contrast, the initial token ID of 'stem' is shared by GPT-4 and Llama 3 (91,994), which suggests that Llama 3 started off with the same tokenizer as GPT-4 but has expanded its vocabulary by adding common Chinese characters as single tokens (along with strings of characters from other scripts), hence its divergence from GPT-4 for the other five characters, which are single Llama 3 tokens. Consistent with this, those single-token characters all have Llama 3 IDs greater than 100,000, so their indices are outside the 100,000-token vocabulary of GPT-4.

Semantic radical	Character	GPT-4 IDs (bytes)	GPT-4o IDs (bytes)	Llama 3 IDs (bytes)
艹 ('grass')	茶 ('tea')	91994, 114 (e88c, b6)	74914 (e88cb6)	108104 (e88cb6)
	茎 ('stem')	91994, 236 (e88c, 8e)	21290, 236 (e88c, 8e)	91994, 236 (e88c, 8e)
口 ('mouth')	吗 (question particle)	7305, 245 (e590, 97)	4447 (e59097)	103054 (e59097)
	吧 (suggestion particle)	7305, 100 (e590, a7)	18208 (e590a7)	102445 (e590a7)
扌 ('hand')	把 ('hold')	24326, 232 (e88c, b6)	32180 (e88cb6)	102178 (e88cb6)
	抱 ('hug')	24326, 109 (e88c, 8e)	105500 (e88c8e)	108623 (e88c8e)

and they invite analogies to morphemic and sub-morphemic category markers in other languages. For example, English nouns and verbs differ subtly in length, stress, and vowel quality, on average (Kelly 1992), and such category markers exist in hundreds of other languages (Contreras Kallens and Christiansen 2020), especially as word endings (e.g., Cutler, Hawkins, and Gilligan 1985). (Note that most Mandarin words comprise multiple characters, so whole characters, along with radicals, sometimes function similarly to suffixes. For example, the words for 'hat', 帽子, and 'mosquito', 蚊子, end with the noun-marking character 子.) Semantic radicals supply different types of information in different characters (e.g., grammatical markers versus conceptual categories), and the link is sometimes obscure (e.g., the character for 'arrogant', 骄, contains the 'horse' radical, 马), but semantic radicals nevertheless influence Chinese character processing in humans (e.g., Feldman and Siok 1999; Taft and Zhu 1997), much like how sub-morphemic cues influence word processing in many other languages (e.g., Dufour 2008; Farmer, Christiansen, and Monaghan 2006; Kwon 2017).

The Chinese script presents a challenge to language models. In terms of pronunciation, phonetic radicals are unreliable cues, in part because, on average over time, written forms persist while spoken forms change. For example, 口 ('mouth') is a

phonetic radical in 加 (‘add’) because historically they had similar pronunciations, but they are now pronounced something like “kau” and “jia” in Mandarin, respectively, and “hau” and “ga” in Cantonese. Moreover, byte-pair encoding tokens derive from UTF-8 encodings, typically with three bytes per Chinese character, and because LLMs perceive combinations of bytes rather than the surface forms of Chinese characters, phonetic radicals are obscured. To overcome these challenges, some Chinese-centric models supplement characters with pinyin (i.e., Romanized phonetic spellings) and stroke or glyph embeddings (i.e., manual or visual information about Character structure; Shi et al. 2015; Si et al. 2023; Sun et al. 2021).

Although UTF-8 encodings obscure phonetic radicals, they imply information about semantic radicals. When Chinese characters were assigned UTF-8 encodings in the early 1990s, they were arranged by semantic radical, so characters that share a semantic radical often begin with the same two bytes. Alternative representations of Chinese characters overlook the information about semantic radicals that comes for free with UTF-8 encodings (e.g., Si et al. 2023; Sun et al. 2021; Wu, Stratos, and Xu 2024). However, the number of characters that contain a radical does not map neatly onto the number of combinations of bytes (i.e., bytes are periodic, but radicals are not), so characters that begin with the same two bytes sometimes contain different semantic radicals, and characters that contain the same radical sometimes begin with different bytes. As illustrated in Figure 1, randomly paired characters share a semantic radical only about 2% of the time, on average, but when pairing characters that begin with the same two bytes, they share a radical about 84% of the time. Crucially, byte-pair encoding produces tokens that often correspond to radicals. Characters that begin with the same GPT-4 token share a semantic radical about 70% of the time, characters that begin with the same GPT-4o token share a semantic radical about 94% of the time, and characters that begin with the same Llama 3 token share a semantic radical about 92% of the time.

The trend in state-of-the-art LLMs is toward longer tokens that more often correspond to words and characters rather than smaller constituents. For example, GPT-4o

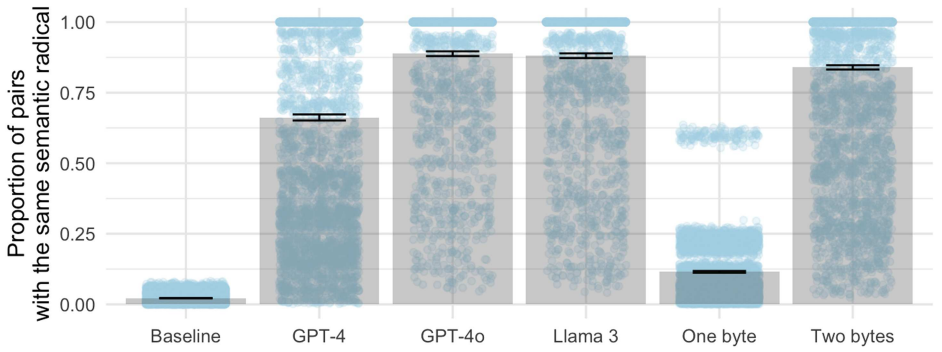

Figure 1
The proportion of pairs of Chinese characters that contain the same semantic radical. The 4,895 characters in the Chinese Lexical Database were each randomly paired with 1,000 other characters (Baseline), paired with 1,000 characters that begin with the same LLM token, or paired with 1,000 characters that begin with the same one or two bytes. There are 4,895 observations in the baseline condition, indicating the proportion of cases where each character shared a semantic radical with its pair, but the LLM token conditions have fewer observations because they exclude characters that comprise a single token and so can be paired only with themselves. Error bars indicate 95% confidence intervals.

has double the vocabulary of GPT-4 (200,000 versus 100,000 tokens), and, as discussed in Table 1, Llama 3 started off with GPT-4's vocabulary and augmented it with Chinese characters (among other tokens). Qwen, a Chinese-centric state-of-the-art open-source LLM, has done something similar (Bai et al. 2023). GPT-4o, Llama 3, and Qwen therefore segment fewer Chinese characters into sub-character tokens than GPT-4 does (and Qwen does so very rarely), which suggests that longer tokens deprive those LLMs of byte-level information about semantic radicals, though again, it remains to be seen whether this affects how LLMs represent meanings.

1.3 The Present Study

Do subword tokens influence how LLMs represent the meanings of words and characters? To answer this question, I conducted six experiments with GPT-4, GPT-4o, and Llama 3.1 (with 70 billion parameters), manipulating whether tokens correspond to semantic radicals or suffixes. Experiment 1 asked the LLMs whether pairs of Chinese characters contain the same semantic radical. If the LLMs associate initial tokens with radicals, they should more often answer "Yes" when pairs begin with the same token. Experiment 2 elicited semantic similarity ratings. If initial tokens influence how LLMs represent the meanings of Chinese characters, they should supply higher ratings for pairs that begin with the same token. Experiment 3 was an odd-one-out task, asking the LLMs which of five characters is unlike the others. If initial tokens influence how LLMs represent characters, they should more often decide that the character with a different initial token is unlike the others. Experiments 4 and 5 compared characters that comprise single tokens (i.e., without subword constituents) to characters that comprise three tokens (i.e., one byte per token), again asking whether pairs of characters contain the same semantic radical (Experiment 4) and conducting an odd-one-out task (Experiment 5). If longer tokens that align with morphemes improve LLM performance, they should identify and categorize one-token characters more accurately than three-token characters. If, on the other hand, sub-morphemic tokens supply LLMs with useful information, they should identify and categorize three-token characters more accurately than they identify and categorize one-token characters.

The imperfect pattern in tokens and semantic radicals makes for a convenient test case to see whether LLMs extract semantic information from subword constituents and whether malformed constituents corrupt word meanings. However, the orthography of Chinese characters is unique, so the findings might not apply to other languages. Experiment 6 therefore comprises one-odd-out tasks in 12 European languages, manipulating whether LLM tokens correspond to inflectional suffixes. If final tokens influence how LLMs represent words, they should more often decide that the word with a different final token is unlike the others. Analyses were conducted in Python and R, using tidyverse (R Core Team 2024; Wickham et al. 2019). All data and scripts are available at `https://osf.io/52cua/`.

2. Experiment 1

Do LLMs "know" about the composition of Chinese characters? That is, can they accurately decide whether pairs of characters contain the same semantic radical, despite not directly perceiving radicals? Do they recognize and utilize correspondences between radicals and initial tokens? And do irregularities in these correspondences lead the LLMs to make faulty inferences? To investigate these issues, Experiment 1 asked GPT-4,

GPT-4o, and Llama 3.1 (with 70 billion parameters) whether pairs of Chinese characters contain the same semantic radical.

2.1 Methods

2.1.1 Materials. I segmented each character in the Chinese Lexical Database (Sun et al. 2018) into GPT-4 tokens and GPT-4o tokens using the tiktoken Python package and into Llama 3 tokens using the transformers package from HuggingFace (Wolf et al. 2020). Of the 4,895 characters, 4,367 comprise multiple GPT-4 tokens (89%), 2,804 comprise multiple GPT-4o tokens (57%), and 3,109 comprise multiple Llama 3 tokens (64%). As shown in Table 2, this experiment manipulates whether pairs of characters share a token and/or radical. The design is 2 (semantic radical: same or different) × 2 (initial token: same or different), with both factors manipulated within items, such that one key character is held constant and is paired with a different target character in each condition. The pair in the +Radical, +Token condition (same semantic radical and same

Table 2
Example items in four conditions. The top row for each LLM contains the key character, which is the same for that item across all conditions. The four rows below contain target characters which are manipulated to share a semantic radical and/or initial token with the key character.

Condition	Character	Radical	Token IDs
GPT-4 key	骆 (camel)	马 (horse)	165, 103, 228
Both (+Token, +Radical)	骁 (good horse)	马 (horse)	165, 103, 223
Token (−Token, +Radical)	鞘 (sheath)	革 (leather)	165, 252, 246
Radical (+Token, −Radical)	驮 (burden of beast)	马 (horse)	77180, 106
Neither (−Token, −Radical)	咛 (instruct)	口 (mouth)	161, 240, 249
GPT-4o key	骆 (camel)	马 (horse)	10151, 228
Both (+Token, +Radical)	骐 (spotted horse)	马 (horse)	10151, 238
Token (+Token, −Radical)	骶 (tailbone)	骨 (bone)	10151, 114
Radical (−Token, +Radical)	驸 (extra horse)	马 (horse)	9722, 116
Neither (−Token, −Radical)	隼 (falcon)	隹 (bird)	11676, 120
Llama 3 key	骆 (camel)	马 (horse)	101805, 228
Both (+Token, +Radical)	骁 (good horse)	马 (horse)	101805, 223
Token (+Token, −Radical)	骷 (skeleton)	骨 (bone)	101805, 120
Radical (−Token, +Radical)	驮 (burden of beast)	马 (horse)	77180, 106
Neither (−Token, −Radical)	矗 (upright)	目 (eye)	100543, 245

initial token) was randomly selected from a list of characters that share both their semantic radical and initial token with the key character. The character closest in log frequency to the +Radical, +Token target was then selected from the −Radical, +Token list, the +Radical, −Token list, and a list comprising 100 randomly selected characters that are −Radical, −Token. For example, the key character 骆 ('camel') and the target 驮 ('the load carried by a pack animal') contain the same semantic radical, 马 ('horse'), but they begin with different GPT-4 tokens, so 驮 is +Radical, −Token. Different tokenizers produce different stimuli: There are 5,144 pairs for GPT-4 (i.e., 1,286 observations per condition), but only 520 pairs for GPT-4o (130 observations per condition) and 816 pairs for Llama 3 (204 observations per condition). GPT-4o has double the vocabulary of tokens that GPT-4 does (roughly 200,000 versus 100,000), so it segments fewer characters into multiple tokens, while Llama 3 seems to have expanded GPT-4's vocabulary to 128,000 tokens by adding common Chinese characters as single tokens (among many other tokens).

Using the OpenAI and Llama APIs, I presented each pair of characters to each LLM in the following prompt, with 骆 ('camel') and 驮 ('load') as examples: *Chinese characters contain semantic radicals. Are "骆" and "驮" Chinese characters? Yes, they are. Do "骆" and "驮" contain the same semantic radical?* The prompt was designed to increase the likelihood that the response (i.e., the next token produced by the LLM) would be "Yes" or "No." The quotation marks ensure that tokenizations of the Chinese characters do not incorporate leading whitespace, as they would not in Chinese text and as the materials were designed. Each pair was also presented with a system prompt: *You are completing a simple task: Decide whether each pair of Chinese characters contains the same semantic radical. Respond only with Yes or No.*

2.1.2 Measures. I measured the likelihood of GPT-4 and GPT-4o responding "Yes" to the above prompt. The logprobs attribute of the chat.completions.create() function in the OpenAI API allows users to record the log probability of an LLM continuing a prompt with a given token. It supplies probabilities for only the 20 most probable tokens, but following the above prompt, "Yes" was always among the 20 most probable tokens. To keep the original probabilities, I set the temperature parameter at the default value of 1. To undo the log-transformation of the original probabilities, I exponentiated them, for a dependent variable between 0 and 1. As discussed below, the pattern of effects is identical when recording whether the next token produced by GPT-4 and GPT-4o is "Yes," rather than extracting the logprobs. The Llama API does not include the logprobs attribute—and at the time of analysis, I did not have the resources necessary for next-token inference with Llama 3.1 70B on a local machine—so I simply recorded whether Llama 3 responded "Yes," again with the default temperature parameter. If the LLMs "know" whether pairs of Chinese characters contain the same semantic radical, the probability of responding "Yes" should be higher in the +Radical conditions. If they associate radicals with initial tokens, the probability should be higher in the +Token conditions. Note that these hypotheses are in no way mutually exclusive (i.e., as reported below, both effects are highly significant).

I also measured the semantic similarity of each pair of characters as represented by FastText. FastText is a machine learning distributional semantic model, like word2vec (Bojanowski et al. 2017; Grave et al. 2018; Mikolov et al. 2013). It represents each character with an embedding that situates that character in high-dimensional semantic space. Characters that occur in similar linguistic contexts tend to have similar meanings, and the cosine similarity of their embeddings indicates their tendency to occur in similar contexts. As mentioned above, FastText embeddings also reflect subword constituents

(i.e., the contexts where a string of letters occurs), but this study uses single Chinese characters, which FastText doesn't decompose to subwords (i.e., its embeddings don't consider bytes or radicals). If LLMs assume that characters with more similar meanings are more likely to share semantic radicals, the likelihood of responding "Yes" should increase as a function of FastText cosine similarity. If the effect of initial tokens were an artefact of characters that share tokens tending to be more semantically similar, for some reason, then the probability of responding "Yes" should not be higher in the +Token condition when accounting for FastText cosine similarity (i.e., when including it as a co-variate in a linear regression model).

2.2 Results and Discussion

Table 3 reports linear regression models for GPT-4, GPT-4o, and Llama 3, regressing the probability of answering "Yes" on three covariates: whether pairs of Chinese characters contain the same semantic radical (sum coded such that +Radical $= 0.5$, $-$Radical $= -0.5$), whether they begin with the same token (sum coded such that +Token $= 0.5$, $-$Token $= -0.5$), and the cosine similarity of their FastText embeddings. The positive effect of shared radicals indicates that all three LLMs identify whether pairs share radicals better than expected by chance (i.e., they're more likely to respond "Yes" when pairs actually share a radical). The positive effect of FastText cosine similarity indicates that the probability of responding "Yes" increases as a function of semantic similarity. Crucially, the positive effect of shared initial tokens indicates that when pairs begin with the same token, the probability of deciding that they contain the same semantic radical increases, consistent with the hypothesis that LLMs extract semantic information from sub-character tokens.

Table 3
Linear regression results of Experiment 1. The probability of GPT-4 and GPT-4o answering "Yes" when asked whether pairs of Chinese characters contain the same semantic radical, and the proportion of "Yes" responses by Llama 3, regressed on whether pairs contain the same radical, whether they begin with the same token, and the cosine similarity of their FastText embeddings. Initial token and semantic radical are sum coded such that same token (+Token) and same radical (+Radical) $= 0.05$, and different token ($-$Token) and different radical ($-$Radical) $= -0.05$.

	B	*SE*	*t*	*p*
GPT-4				
Intercept	0.566	0.036	15.6	< 2e-16
Initial token	0.224	0.010	21.4	< 2e-16
Semantic radical	0.568	0.011	51.6	< 2e-16
FastText similarity	0.397	0.056	7.0	2e-12
GPT-4o				
Intercept	0.566	0.036	15.6	< 2e-16
Initial token	0.542	0.032	17.2	< 2e-16
Semantic radical	0.365	0.032	11.3	< 2e-16
FastText similarity	0.290	0.145	2.0	.0456
Llama 3				
Intercept	0.594	0.039	15.2	< 2e-16
Initial token	0.434	0.032	13.5	< 2e-16
Semantic radical	0.187	0.034	5.5	5e-8
FastText similarity	0.471	0.163	2.9	.0041

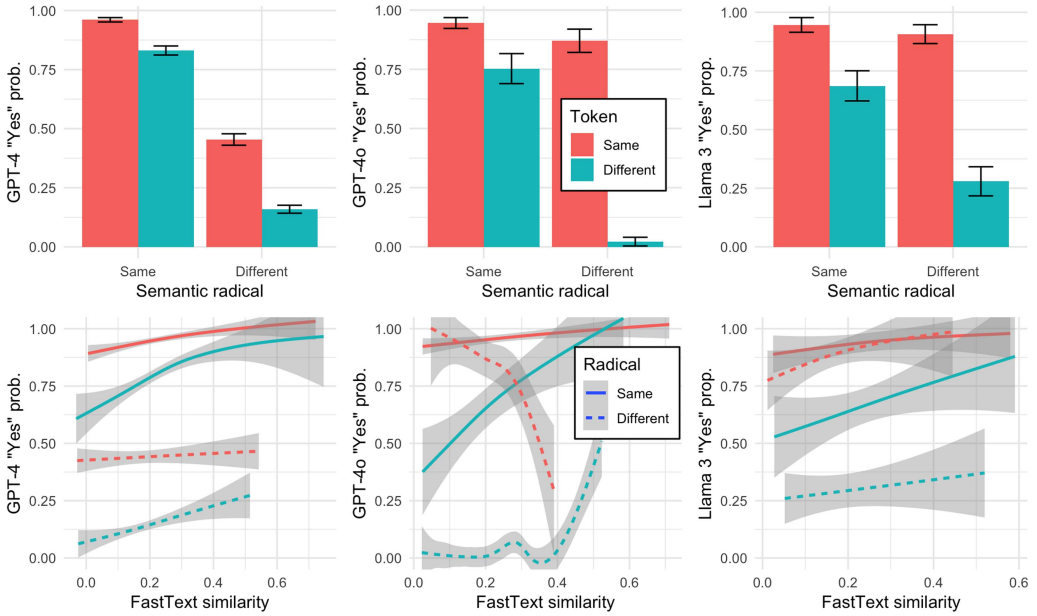

Figure 2
The impact of initial tokens, semantic radicals, and FastText similarity in Experiment 1. The *y*-axes indicate the probability of answering "Yes," for GPT-4 and GPT-4o, or the proportion of times answering "Yes," for Llama 3, when asked whether Chinese characters contain the same semantic radical. In the bottom row, the *x*-axes indicate the cosine similarity of Chinese characters' FastText embeddings. The smoothed lines (using GAM) are for illustration only. Shaded bands and and error bars indicate 95% confidence intervals.

Notice how, in the top row of Figure 2, the LLMs are very likely to respond "Yes" in the +Token, +Radical condition and very unlikely to respond "Yes" in the −Token, −Radical condition, which demonstrates accurate confidence about the composition of Chinese characters in those conditions. But initial tokens interfere with this confidence and accuracy. In the +Radical, −Token condition, the probability of responding "Yes" decreases relative to the +Radical, +Token condition, and in the −Radical, +Token condition, the probability increases relative to the −Radical, −Token condition. Similarly, the large differences between the −Token, +Radical condition and the −Token, −Radical condition demonstrates that GPT-4, GPT-4o, and Llama 3 clearly have some "knowledge" of semantic radicals, independent of tokens (and independent of distributional semantic similarity), which presumably derives from explicit references to radicals in training data, but the orthogonal manipulation of shared tokens demonstrates that, despite this knowledge, they are strongly influenced by shared initial tokens, leading to many false alarms.

The impact of shared initial tokens is most striking in the +Token, −Radical condition for GPT-4o and Llama 3: They are more likely to respond "Yes" when pairs share tokens but not radicals than when they share radicals but not tokens. The differences between those conditions are significant in paired t-tests: for GPT-4o, $t(129) = 2.8$, $p = .006$; for Llama 3, $t(203) = 5.8$, $p = 2e\text{-}8$. GPT-4o and Llama 3 segment fewer characters into multiple tokens than GPT-4 does, and because byte-pair encoding is based on frequency, those characters tend to be rare. (As measured by the Chinese

Lexical Database, the GPT-4 stimuli comprise more common characters than the GPT-4o or Llama 3 stimuli do; see the online supplement.[1]) So, this larger impact of shared tokens on GPT-4o and Llama 3 likely reflects a difference in stimuli from GPT-4, not that GPT-4o and Llama 3 are more susceptible to token-based inferences than GPT-4 is.

The bottom row of Figure 2 illustrates how the effect of shared initial tokens is not due to characters being more semantically similar in the +Token condition. For a given similarity value, the predicted probability of responding "Yes" varies depending on whether the pair begins with the same token. This effect is most pronounced for pairs with dissimilar FastText embeddings, such that the probability of responding "Yes" decreases substantially only for pairs that don't share a token. As reported in the online supplement, the interaction of FastText similarity with shared initial tokens has a negative coefficient, confirming that the positive relationship between FastText similarity and "Yes" probability (i.e., being less likely to decide "Yes" for characters that have less similar meanings) is mitigated when characters begin with the same token (p = 6e-7 for GPT-4, p = 3e-9 for GPT-4o, p = .008 for Llama 3). The high probability of deciding that semantically dissimilar characters share semantic radicals when they share initial tokens suggests that shared initial tokens influence how GPT-4, GPT-4o, and Llama 3 represent the meanings of Chinese characters. However, these judgments only indirectly address that question, and prompts that ask about sub-character constituents could encourage the LLMs to attend to subword tokens, inflating what might otherwise be a negligible effect.

3. Experiment 2

Experiment 1 demonstrated that GPT-4, GPT-4o, and Llama 3 respond far more accurately than expected by chance when asked whether pairs of Chinese characters contain the same semantic radical. However, their accuracy depends largely on those characters sharing initial tokens. The impact of shared tokens is most pronounced in characters that are semantically dissimilar, which suggests that shared tokens inflate the perceived similarity of those characters to LLMs. This experiment more directly investigated that issue by asking the LLMs to rate the semantic similarity of characters, without referring to radicals.

3.1 Methods

3.1.1 Materials. This experiment used the same pairs of characters as in Experiment 1, with the same design of 2 (semantic radical: same or different) × 2 (initial token: same or different), manipulated within items. Using the OpenAI and Llama APIs, I presented each pair to each LLM in the following prompt, again with 骆 ('camel') and 驮 ('load carried by a pack animal') as examples: *On a scale of 1 to 5, with 1 meaning totally unrelated and 5 meaning very similar, how semantically similar are the Chinese characters "骆" and "驮"?*. Each pair was also presented with a system prompt: You are completing a simple task: *Rate the semantic similarity of Chinese characters. Respond only with a number between 1 and 5.*

3.1.2 Measures. As shown in Table 4, I measured the likelihood of GPT-4 and GPT-4o responding to the above prompt with 1, 2, 3, 4, and 5, using the logprobs attribute of the

1 The online supplement is available at `https://osf.io/52cua/`.

Table 4
Probabilistic semantic similarity rating from GPT-4 for one item. "Product" refers to the rating value multiplied by the SoftMax-transformed probability. These values are in response to the following prompt: *On a scale of 1 to 5, with 1 meaning totally unrelated and 5 meaning very similar, how semantically similar are the Chinese characters* "骆" and "驮"? ('camel' and 'load').

Rating value	Log prob.	SoftMax prob.	Product
1	−7.87	.000	0.00
2	−5.04	.006	0.01
3	−2.88	.056	0.17
4	−0.44	.645	2.58
5	−1.23	.292	1.46
Summed rating			4.22

chat.completions.create() function, with the temperature parameter at the default value of 1. That is, I recorded five logprobs per prompt. I then used SoftMax, from Scikit-learn (Pedregosa et al. 2011), to transform them such that the probabilities for the five valid ratings summed to 1 for each item in each condition. When presenting GPT-4 with the example prompt, the most probable rating is 4, with a logprob of −0.4 and a SoftMax probability of 0.65. (Recall that the log of 1 is 0, so negative log probabilities close to zero indicate high probability.) I multiplied each rating value (e.g., 4) by its SoftMax probability (e.g., 0.65) and then summed all five such products of rating value and SoftMax probability, for a dependent variable between 1 and 5. This summed rating is equivalent to prompting the LLM many times, recording only valid ratings, and measuring the mean rating. In this example, the summed rating is 4.22, reflecting that the most probable response is 4 and the next most probable is 5, with low probabilities for 1, 2, and 3. As reported in the online supplement, the pattern and significance of effects are identical if instead simply recording the LLMs' responses to each prompt (in the above example, a rating of 4). The Llama API does not include the logprobs attribute, so I recorded the numerical value of the first token produced by Llama 3. In 813 of the 816 Llama 3 trials, the next token was a number between 1 and 5.

If the LLMs provide reasonable ratings, then ratings should increase as a function of the cosine similarity of FastText embeddings. That is, characters with more similar meanings should receive higher ratings. If shared initial tokens influence how the LLMs represent the meanings of Chinese characters, ratings should be higher in the +Token conditions.

3.2 Results and Discussion

Table 5 reports three linear regression models, each regressing semantic similarity ratings on three covariates, as in Experiment 1: whether pairs of Chinese characters contain the same semantic radical, whether they begin with the same token, and the cosine similarity of their FastText embeddings. The positive effect of FastText cosine similarity indicates that ratings increase as a function of semantic similarity. The positive effect of shared semantic radicals indicates that all three LLMs rate pairs as more similar in meaning when they share semantic radicals. Crucially, the positive effect of shared initial tokens indicates that when pairs begin with the same token, the LLMs give higher semantic similarity ratings, consistent with the hypothesis that initial

Table 5
Linear regression results of Experiment 2. Probabilistic semantic similarity ratings from GPT-4 and GPT-4o for pairs of Chinese characters, and numerical responses from Llama 3, regressed on whether pairs contain the same radical, whether they begin with the same token, and the cosine similarity of their FastText embeddings. Initial token and semantic radical are sum coded such that same token (+Token) and same radical (+Radical) = 0.05, and different token (−Token) and different radical (−Radical) = −0.05.

	B	*SE*	*t*	*p*
GPT-4				
Intercept	1.010	0.010	112.0	< 2e-16
Initial token	0.592	0.008	70.3	< 2e-16
Semantic radical	0.592	0.008	28.2	< 2e-16
FastText similarity	1.889	0.043	43.8	< 2e-16
GPT-4o				
Intercept	1.031	0.026	40.2	< 2e-16
Initial token	0.264	0.022	11.9	< 2e-16
Semantic radical	0.546	0.023	24.0	< 2e-16
FastText similarity	2.330	0.102	22.8	< 2e-16
Llama 3				
Intercept	1.882	0.102	18.5	< 2e-16
Initial token	0.744	0.084	8.9	< 2e-16
Semantic radical	0.723	0.088	8.2	1e-15
FastText similarity	2.563	0.424	6.0	3e-9

tokens influence how Chinese characters are represented and inflate the similarity of those representations.

The bottom row of Figure 3 illustrates the positive relationship between FastText similarity and semantic similarity ratings, which demonstrates that the LLMs supply reasonable ratings. Notably, the slope is much steeper in the +Radical conditions than in the −Radical conditions, reflected in a positive interaction of FastText similarity with shared semantic radicals, as reported in the online supplement ($p < 2e\text{-}16$ for GPT-4; $p = 3e\text{-}13$ for GPT-4o; $p = .022$ for Llama 3). This is not directly relevant to the hypothesis about the impact of initial tokens on the representations of meanings, but it does suggest that semantic radicals are salient to LLMs, which underscores the problem that motivates this study: By processing text as a sequence of tokens, LLMs lack direct access to information in word form, and the impact of semantic radicals on semantic similarity ratings suggests that GPT-4, GPT-4o, and Llama 3 appreciate the value of that information. As demonstrated in Experiment 1, LLMs use initial tokens to identify radicals, so when they use semantic radicals to represent meanings, they will sometimes be misled. The crucial finding, again, is that shared initial tokens inflate semantic similarity ratings, as is clear in the top row of Figure 3.

4. Experiment 3

Experiment 2 demonstrated that shared initial tokens influence how GPT-4, GPT-4o, and Llama 3 represent the meanings of Chinese characters. Even though the prompt in Experiment 2 made no reference to semantic radicals or subword constituents, tokens impacted representations. Furthermore, semantic radicals strongly influenced semantic similarity ratings, which suggests that despite not directly perceiving semantic radicals,

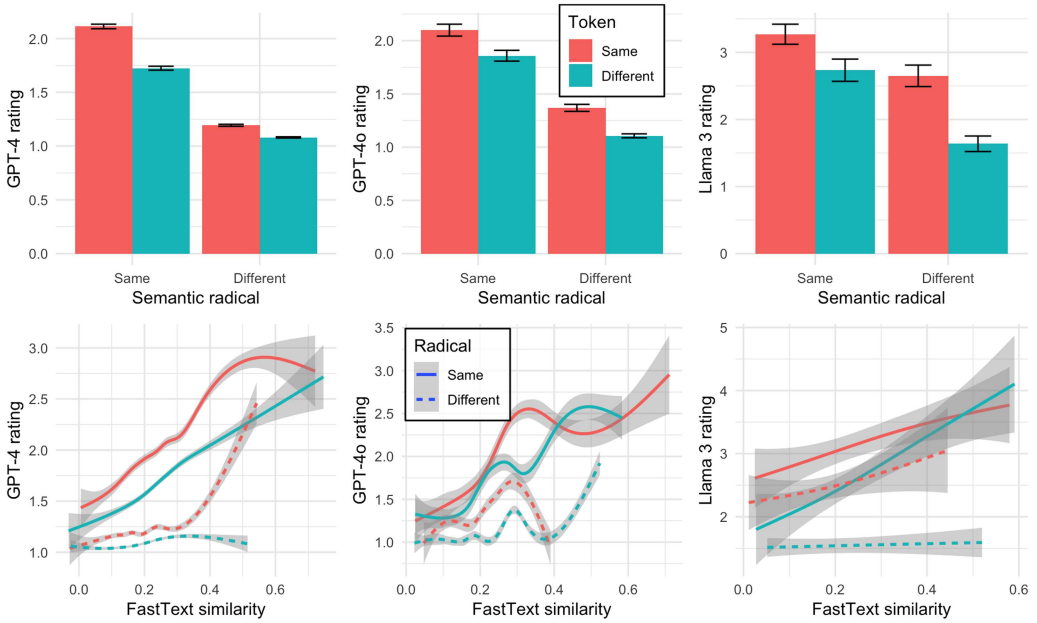

Figure 3
The impact of initial tokens, semantic radicals, and FastText similarity in Experiment 2. The *y*-axes indicate semantic similarity ratings for pairs of Chinese characters. In the bottom row, the *x*-axes indicate the cosine similarity of characters' FastText embeddings. The smoothed lines (using GAM) are for illustration only. Shaded bands and error bars indicate 95% confidence intervals.

the LLMs appreciate their salience to Chinese characters. This experiment therefore investigated whether LLMs are inclined to categorize Chinese characters according to shared radicals and whether the imperfect correspondence between radicals and tokens interferes with their ability to do so. In an odd-one-out task, I asked the LLMs which one of five characters is unlike the others.

4.1 Methods

4.1.1 Materials. This experiment built on the stimuli from Experiments 1 and 2. Each prompt in those experiments presented a pair of Chinese characters. In this odd-one-out task, I randomly selected three more characters that share both a semantic radical and an initial token with the key character. For example, the characters 骆,骏,骝,, and 骄 ('camel', 'good horse', 'reddish horse', and 'arrogant') all contain the 'horse' radical and begin with the same GPT-4 token. That group of four was then presented along with the target in a given condition, such as 驮 ('the load carried by a pack animal'), which contains the same semantic radical but begins with a different token (i.e., +Radical, −Token). The design is again 2 (semantic radical: same or different) × 2 (initial token: same or different), both manipulated within items. Not all items from Experiments 1 and 2 have enough characters to make groups of five (i.e., they lack three additional +Token, +Radical characters), so GPT-4 has 1,115 items in Experiment 3 rather than

1,286, GPT-4o has 75 items rather than 130, and Llama 3 has 134 items rather than 204. Again using the OpenAI and Llama APIs, I presented each group of five characters in the following prompt (with English glosses included for reference only):

Which one of these five Chinese characters is not like the other four?

A: "骆" (camel)

B: "骏" (good horse)

C: "骝" (reddish horse)

D: "骄" (arrogant)

E: "驮" (load)

The order of characters was randomized for each item. The four +/−Radical, +/−Token targets in each item were presented in the same position in four separate prompts (i.e., in the above example, all would be presented in position E). Each group of five was also presented with a system prompt: *You are completing a simple task: Decide which one of five Chinese characters is not like the other four. Respond only with one of the following letters: A, B, C, D, E.*

4.1.2 Measures. Similar to Experiment 2, I measured the likelihood of GPT-4 and GPT-4o responding to the above prompt with A, B, C, D, or E, using the logprobs attribute of the chat.completions.create() function, with the temperature parameter set at the default value of 1. I again used SoftMax to transform the logprobs such that the probabilities for the five valid responses summed to 1 for each item in each condition. I then recorded the SoftMax-transformed probability of the target character, for a dependent variable between 0 and 1. In the above example, the probability of GPT-4 deciding that 驮 ('the load carried by a pack animal') is the odd one out is less than .001. The Llama API does not include the logprobs attribute, so I simply recorded the next token produced by Llama 3. In 532 of 536 Llama 3 trials, the next token was a single letter between A and E. For GPT-4 and GPT-4o, the pattern of effects is identical when recording the next token rather than the logprobs.

To quantify the semantic similarity of a character to the rest of the group, I calculated the mean cosine similarity of the target character's FastText embedding to the other four characters' embeddings. For example, 驮 ('load') has a mean cosine similarity of .273 to 骆 ('camel'), 骏 ('good horse'), 骝 ('reddish horse'), and 骄 ('arrogant'). I then divided that value by the group mean (i.e., the mean similarity of each character to the other four characters). For example, the mean cosine similarity of the other characters is .281, and .281 divided by .273 gives 驮 a relative FastText similarity of 0.97. This value is less than one, which indicates that it is less semantically similar to the other characters than they are to each other, on average. If the most semantically dissimilar character is the most likely to be the odd one out, then a character that is high in relative similarity to the others is unlikely to be the odd one out. That is, odd-one-out probability should decrease as a function of relative FastText similarity. If the LLMs recognize that four or five of the characters share a semantic radical, and if radicals are salient, then the probability of a target being the odd one out should decrease in the +Radical conditions. If the LLMs conflate shared initial tokens with shared semantic radicals, the probability of a target being the odd one out should decrease in the +Token conditions.

4.2 Results and Discussion

Table 6 reports three linear regression models, regressing the probability of a target being the odd one out on three covariates, as in Experiments 1 and 2: whether target characters contain the same semantic radical as the other group members, whether they begin with the same token as the other group members, and the relative FastText similarity of the target to the other group members. The negative effect of shared semantic radicals indicates that all three LLMs are less likely to decide that a target is the odd one out if it shares a semantic radical with all other group members. Crucially, the negative effect of shared initial tokens indicates that when a target begins with the same token as all other group members, the LLMs are less likely to decide that it is the odd one out, consistent with the hypothesis that initial tokens influence how the LLMs represent Chinese characters and that this is salient when categorizing characters.

The negative effect of relative FastText similarity indicates that the LLMs are less likely to decide that a target is the odd one out when it is more semantically similar to the other group members, relative to their similarity to each other, as illustrated in the bottom row of Figure 4. Similar to Experiment 1, the likelihood of GPT-4o and Llama 3 deciding that the target character is the odd one out is lower in the +Token, −Radical condition than in the −Token, +Radical condition, as illustrated in the top row of Figure 4. In t-test, $p = .007$ for GPT-4o, $p = .043$ for Llama 3. As in Experiment 1, this likely reflects the fact that the GPT-4o and Llama 3 stimuli comprise lower frequency characters than the GPT-4 stimuli do. The crucial result is that shared initial tokens have a substantial impact on LLM performance, even when accounting for shared semantic radicals and distributional semantic similarity.

Table 6
Linear regression results of Experiment 3. The probability of selecting a target character as the odd one out of a group of five, regressed on whether it contains the same radical as the other members, whether it begins with the same token as the other members, and the mean FastText similarity of the target character (+/−Token and +/−Radical) to the four other characters in that group divided by the mean cosine similarity of those characters to each other. Initial token and semantic radical are sum coded such that same token (+Token) and same radical (+Radical) = 0.05, and different token (−Token) and different radical (−Radical) = −0.05.

	B	*SE*	*t*	*p*
GPT-4				
Intercept	0.895	0.022	40.7	< 2e-16
Initial token	−0.294	0.011	−27.7	< 2e-16
Semantic radical	0.448	0.011	−39.1	< 2e-16
FastText similarity	−0.218	0.023	−9.4	< 2e-16
GPT-4o				
Intercept	0.903	0.078	11.6	< 2e-16
Initial token	−0.455	0.040	−11.3	< 2e-16
Semantic radical	−0.218	0.043	−5.1	7e-7
FastText similarity	−0.271	0.084	−3.2	.0014
Llama 3				
Intercept	0.666	0.082	8.1	6e-15
Initial token	−0.425	0.040	−10.5	< 2e-16
Semantic radical	−0.239	0.045	−5.3	2e-7
FastText similarity	−0.192	0.091	−2.1	.0343

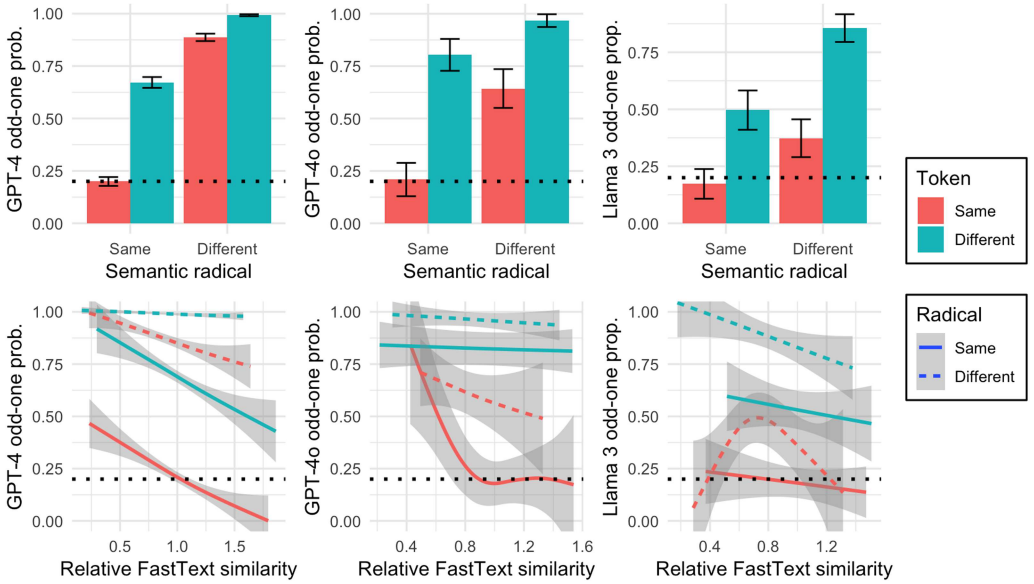

Figure 4
The impact of initial tokens, semantic radicals, and FastText similarity in Experiment 3. The *y*-axes indicate the likelihood of responding that the target character is the odd one out of a group of five. In the bottom row, the *x*-axes indicate the mean cosine similarity of the target character to the four other characters in that group divided by the mean cosine similarity of those four other characters to each other. The dotted black lines indicate the at-chance level (i.e., a .2 probability of selecting the target as the odd one out of five). The smoothed lines (using GAM) are for illustration only. Shaded bands and error bars indicate 95% confidence intervals.

5. Experiment 4

Experiments 1, 2, and 3 provide evidence that GPT-4, GPT-4o, and Llama 3 recognize and utilize correspondences between initial tokens and semantic radicals in Chinese characters. The problem is that byte-pair encoding sometimes distorts those constituents. One solution might be larger vocabularies of tokens that segment fewer words into subwords, as is the trend in LLMs. Recall that GPT-4o segments only 57% of the 4,895 characters in the Chinese Lexical Database into multiple tokens, compared to 89% for GPT-4. However, whole-word tokens obscure what subword tokens distort, which is to say, whole-word tokens supply LLMs with even less information about patterns in bytes. This experiment therefore investigated whether LLMs identify semantic radicals more or less accurately in characters that comprise one token than in characters that comprise three single-byte tokens.

5.1 Methods

This experiment followed the same procedure as Experiment 1, except it included only characters that comprise one three-byte token or characters that comprise three single-byte tokens. It has a design of 2 (semantic radical: same or different) × 2 (number of tokens: one or three). Semantic radical is manipulated within items, and number of tokens is manipulated between items. The pair in the +Radical condition was randomly selected from a list of characters that share their semantic radical with the key character.

The character closest in log frequency to the +Radical target was then selected from a list comprising 100 randomly selected characters that are −Radical. For example, 时 ('time') shares the 'sun' radical (日) with 星 ('star') and doesn't share a radical with 十 ('ten'). There are 946 one-token pairs for GPT-4 (473 observations per condition), 2,230 three-token pairs for GPT-4 (1,115 per condition), 4,078 one-token pairs for GPT-4o (2,039 per condition), and only 34 three-token pairs for GPT-4o (17 per condition). Llama 3 segments only three of the 4,895 characters in the Chinese Lexical Database into single-byte tokens, so it has no valid items and is excluded from Experiments 4 and 5.

I did not manipulate whether pairs begin with the same token, because doing so leaves no items in the three-token conditions (and in the one-token condition, characters cannot share tokens). That is, the vast majority of characters that contain the same semantic radical begin with the same byte, which precludes a within-item manipulation of both radical and token in three-token characters. The inverse is not true, though: Most characters that begin with the same byte contain different semantic radicals. Recall, as illustrated in Figure 1, how characters that begin with the same byte share a semantic radical only 11% of the time, on average, so bytes are most informative when combined. The point is that any advantage for three-token characters over one-token characters could be attributed to their sharing tokens. I see this not as a confound but as inherent to value of single-byte tokens, which convey information about word form that larger tokens obscure. (Moreover, it could be a confound only if you accept the hypothesis that sub-character tokens influence how LLMs represent Chinese characters.)

The prompt (*Do "时" [time] and "星" [star] contain the same semantic radical?*) and dependent variable (the probability of GPT-4 and GPT-4o answering "Yes") are identical to Experiment 1. If whole-character tokens obscure important information in word form, GPT-4 and GPT-4o should be less accurate at identifying pairs that share radicals in the one-token condition than in the three-token condition. However, LLMs are powerful black boxes, and they have some knowledge of sub-token constituents (Edman, Schmid, and Fraser 2024; Itzhak and Levy 2022; Kaushal and Mahowald 2022), so it's entirely possible that the LLMs will perform better on one-token characters. In fact, one-token characters are much higher in frequency than three-token characters, on average (like the difference in stimuli for GPT-4 versus GPT-4o and Llama 3, discussed in Experiments 1 and 3; see the online supplement). If GPT-4 and GPT-4o are more familiar with higher frequency characters, and if those LLMs rely on knowledge of characters implicit in training data (rather than relying on tokens as superficial cues), they should identify one-token pairs that share radicals more accurately than they identify three-token characters.

5.2 Results and Discussion

As illustrated in Figure 5, both GPT-4 and GPT-4o are far more likely to decide that characters contain the same semantic radical in the +Radical condition than in the −Radical condition. Crucially, they are more likely to correctly decide "Yes" when +Radical characters comprise three one-byte tokens than when they comprise a single token. For GPT-4, the difference between the +Radical, three-token condition (mean probability = .986) and the +Radical, one-token condition (mean = .219) is highly significant in an unpaired t-test: $t(1{,}586) = 44.3$, $p < 2e\text{-}16$. For GPT-4o, the difference between the +Radical, three-token condition (mean = .999) and the +Radical, one-token condition (mean = .432) is again highly significant: $t(2{,}054) = 65.4$, $p < 2e\text{-}16$. The pessimistic conclusion is that large vocabularies of single-token words obscure valuable

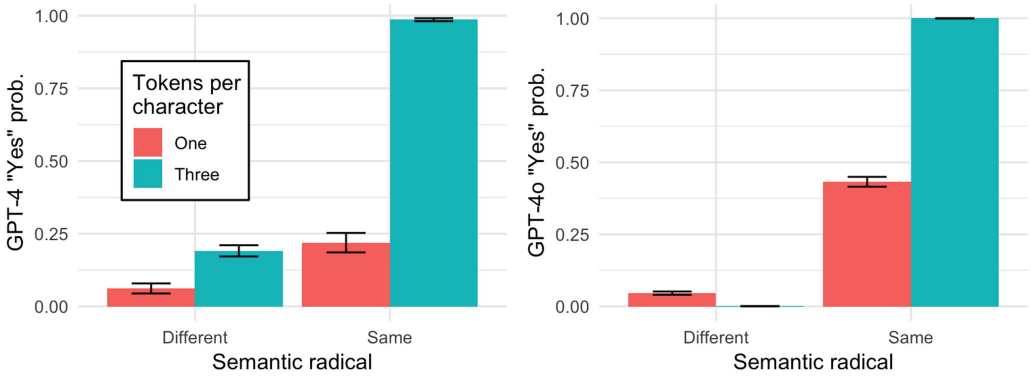

Figure 5
The impact of whole-character tokens versus single-byte tokens in Experiment 4. The *y*-axes indicate the probability of GPT-4 and GPT-4o responding "Yes" when asked whether pairs of Chinese characters contain the same semantic radical. Both LLMs are more likely to correctly answer "Yes" in the same-Radical condition when characters comprise three tokens than when they comprise one token. Error bars indicate 95% confidence intervals.

information in word form and so are unlikely to be a satisfactory solution to the problem of LLMs drawing bad inferences from malformed tokens.

6. Experiment 5

Experiment 4 provides evidence that GPT-4 and GPT-4o are worse at identifying semantic radicals in Chinese characters that comprise one token than in characters that comprise three single-byte tokens. However, that prompt explicitly referred to semantic radicals, which may have encouraged the LLMs to attend to subword constituents. Experiment 5 therefore tested performance in another odd-one-out task, without mentioning radicals.

6.1 Methods

This experiment followed the same procedure as Experiment 3, except it included only characters that comprise one token or characters that comprise three tokens. As in Experiment 4, it has a design of 2 (semantic radical: same or different) $\times$ 2 (number of tokens: one or three). Semantic radical is manipulated within items, and number of tokens is manipulated between items. As in Experiment 3, I built on the materials from Experiment 4 by adding three more characters that share a semantic radical with the key character to each item, for groups of five. There are 624 one-token groups for GPT-4 (312 observations per condition), 2,128 three-token groups for GPT-4 (1,064 per condition), 3,702 one-token groups for GPT-4o (1,851 per condition), and only 18 three-token groups for GPT-4o (9 per condition). The prompt (*Which one of these five Chinese characters is not like the other four?*) and dependent variable (the probability of selecting the target character as the odd one out) are identical to Experiment 3. If GPT-4 and GPT-4o rely on sub-character tokens to identify semantic radicals, and assuming that shared semantic radicals are salient to the categorization of characters, then they should be more likely to decide that the character with a different semantic radical from the

other group members is the odd one out in the three-token condition than in the one-token condition.

6.2 Results and Discussion

As illustrated in Figure 6, both GPT-4 and GPT-4o are more likely to decide that targets in the −Radical condition are the odd one out, compared to targets in the +Radical condition. Crucially, they are more likely to decide that the −Radical target is the odd one out when characters comprise three single-byte tokens than when they comprise one token. For GPT-4, the difference between the −Radical, three-token condition (mean probability = .987) and the −Radical, one-token condition (mean = .409) is highly significant in an unpaired t-test: $t(1{,}374) = 22.6$, $p <$ 2e-16. For GPT-4o, the difference between the −Radical, three-token condition (mean = .998) and the −Radical, one-token condition (mean = .589) is also highly significant, despite the small number of observations in the three-token condition: $t(1{,}854) = 43.9$, $p <$ 2e-16. This provides further evidence that LLMs appreciate the importance of semantic radicals when categorizing characters and that whole-character tokens make semantic radicals harder to identify.

7. Experiment 6

Experiments 3 and 5 provided evidence that GPT-4, GPT-4o, and Llama 3 categorize Chinese characters according to semantic radicals, but to do so, they rely on sub-character tokens, and when tokens don't correspond to radicals, the LLMs miscategorize characters. This suggests that tokenization creates discrepancies in how LLMs and humans represent meanings, impeding our ability to instruct and interpret LLMs. Chinese characters provide a convenient test case for this problem, but Chinese orthography is unique, which calls into question whether these results will generalize to LLM representations more broadly. Category markers are not unique, though; suffixes serve

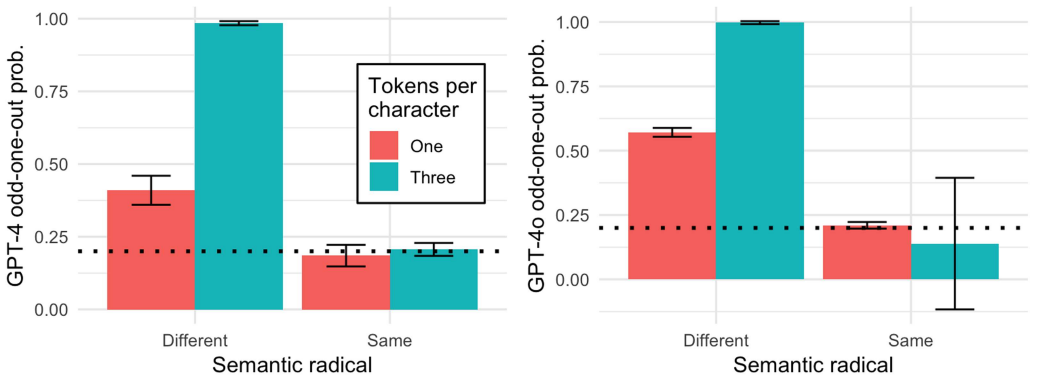

Figure 6
The impact of whole-character tokens versus single-byte tokens in Experiment 5. The *y*-axes indicate the probability of GPT-4 and GPT-4o deciding that the target character is the odd one out of a group of five characters that contain the same semantic radical. Both LLMs are more likely to select the character with a different radical when characters comprise three tokens than when they comprise one token.The dotted black lines indicate the "at chance" level (i.e., a 0.2 probability of selecting the target as the odd one out of five). Error bars indicate 95% confidence intervals.

Table 7
Number of items in Experiment 6 for twelve languages. "Llama 3 exclusions" refers to responses that were not single letters from A to E. "Llama 3" refers to remaining valid responses.

Language	GPT-4	GPT-4o	Llama 3	Llama 3 exclusions
Catalan	3,120	4,014	2,929	251
Czech	5,574	4,128	4,491	216
English	1,200	1,266	1,168	32
Finnish	201	33	202	8
French	4,731	4,560	4,443	351
German	579	414	612	30
Hungarian	939	804	979	38
Italian	4,140	2,790	3,845	211
Portuguese	3,606	4,329	3,401	232
Russian	7,083	2,220	2,273	121
Spanish	2,352	2,808	2,222	148
Swedish	548	651	492	21
Total	34,073	28,017	27,057	1,659

this function in a wide variety of languages (e.g., Cutler, Hawkins, and Gilligan 1985). So, in another odd-one-out task, Experiment 6 investigates whether tokens influence categorization in 12 European languages (10 Indo-European and 2 Uralic—namely, Finnish and Hungarian); see Table 7.

7.1 Methods

This experiment tested the hypothesis that GPT-4, GPT-4o, and Llama 3 treat final tokens like inflectional suffixes, similar to how they treat initial tokens like semantic radicals. It investigates the twelve languages which are parsed into inflectional morphemes in the Morphynet database (Batsuren, Bella, and Giunchiglia 2021) and which have word frequency data available from wordfreq (Speer 2022). For each language, I kept only words that are among the 50,000 most common according to wordfreq, which have inflectional suffixes according to Morphynet, and for which that suffix occurs in the surface form of the word. I segmented each word into GPT-4 tokens, GPT-4o tokens, and Llama 3 tokens. For each language using each LLM, I randomly selected groups of four words which end with the same suffix and for which that suffix is the final token in the word, and which are identical in number of tokens, length in bytes, part of speech (noun, verb, or adjective), and frequency range (namely, the same frequency decile, grouping the words in each language into 10 bins). Then, I randomly selected one target word that meets all the same criteria (the same token condition), one that meets the same criteria but ends with a different token (the different token condition), and one that meets the same criteria but ends with a different suffix and a different token (the different suffix condition).

The prompt is essentially the same as Experiments 3 and 5, as seen in two GPT-4 example items in the different token condition, in French and English. In French, the first four words end with the same suffix ("-ons", which marks plural first-person verbs)

and with the same GPT-4 token (4239, "ons"), while in this example, the target word ends with a different token (26692, "rons"):

Which one of these five French words is not like the other four?

A: portons (carry)

B: sortons (exit)

C: perdons (lose)

D: nichons (nest)

E: entrons (enter)

In English, the first four words end with GPT-4 token 287 ("ing"), and the target word ends with token 61818 ("pering"):

Which one of these five English words is not like the other four?

A: fermenting

B: professing

C: captioning

D: enveloping

E: whimpering

As in Experiments 3 and 5, the order of the target is randomized for each item and kept constant across conditions within that item. I measured the likelihood of GPT-4 and GPT-4o responding to the above prompt with A, B, C, D, or E, and I recorded the next token produced by Llama 3. In 27,057 of 28,716 Llama 3 trials, the next token was a single letter from A to E. As discussed below, the pattern of effects is identical when measuring the next token produced by GPT-4 and GPT-4o, rather than logprobs. If the LLMs recognize that four or five of the words share a suffix, then the probability of a target being the odd one out should be greatest in the different suffix condition. If the LLMs conflate final tokens with suffixes, the probability of a target being the odd one out should be greater in the different token condition than in the same token condition.

7.2 Results and Discussion

As illustrated in Figure 7, all three LLMs are most likely to decide that a target in the different suffix condition is the odd one out. GPT-4 and GPT-4o decide so over 90% of the time, and Llama 3 decides so about 60% of the time. All three LLMs are right around the at-chance level (20%) of deciding that a target in the same token condition is the odd one out. Crucially, all three LLMs are more likely to decide that a target in the different token is the odd one out, compared to the same token condition: 34% of the time for GPT-4 and GPT-4o, and 26% for Llama 3. These differences are highly significant in t-tests (paired for GPT-4 and GPT-4o, unpaired for Llama 3 because exclusions lead to a different number of observations per condition): for GPT-4, t (11,357) = 31.0, $p <$ 2e-16; for GPT-4o, t (9,338) = 30.2, $p <$ 2e-16; for Llama 3, t (17,841) = 7.4, $p =$ 1e-13. When conducting separate t-tests for each language, reported in the online supplement, the difference between the same token and different token conditions fails to reach significance in Hungarian for GPT-4, in Finnish and Swedish for GPT-4o, and in

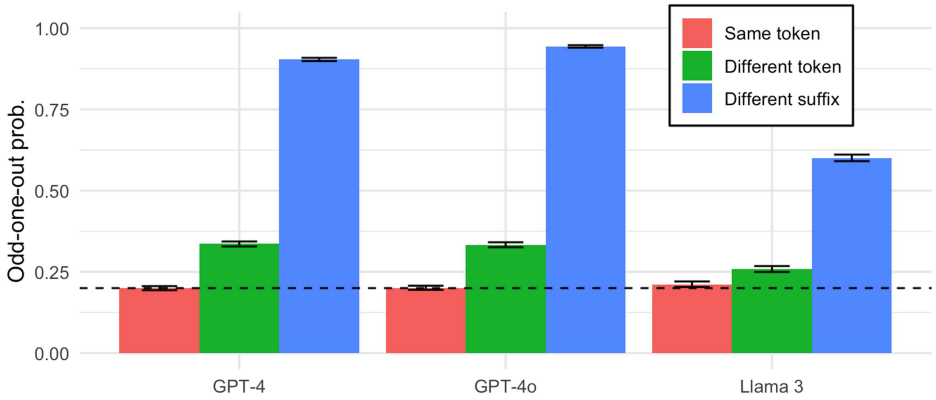

Figure 7
The impact of suffixes and final tokens in Experiment 6. The *y*-axes indicate the likelihood of responding that the target word is the odd one out of a group of five. The dotted black line indicates the at-chance level (i.e., a .2 probability of selecting the target as the odd one out of five). The error bars indicate 95% confidence intervals.

Finnish, French, German, Hungarian, and Swedish for Llama 3. Notice that in all cases, the difference is significant for English, indicating that all three LLMs rely on subword tokens to categorize English words, despite (presumably) ample exposure to English in their (proprietary) training data.

The smaller and less consistent effects in Llama 3 are not a consequence of the different measure used for Llama 3. When investigating the next token produced by GPT-4 and GPT-4o (rather than the underlying probability of selecting the target token), the same languages are significant, and the mean probability for each condition is almost identical to those reported above. The smaller difference between the same token and different token conditions in Llama 3 could indicate that it is less susceptible to misleading tokens, but recall the substantially lower likelihood of Llama 3 deciding that a target in the different suffix condition is the odd one out (60%). Along with Llama 3 correctly avoiding false alarms (i.e., spuriously deciding that targets in the different token condition are the odd one out less often than GPT-4 and GPT-4o do), Llama 3 is also guilty of more misses (i.e., failing to identify targets in the different suffix condition). If you accept decisions to exclude targets in the different suffix condition as "correct," then the d' score (standardized hit rate minus standardized false alarm rate) for GPT-4 and GPT-4o is 1.93 and 2.20, respectively, compared to 0.98 for Llama 3.

Overall, Experiment 6 demonstrates that the effects observed in Chinese generalize to European languages. LLMs treat tokens like linguistic units, which is to say, the process of detokenization (cf. Elhage et al. 2022; Kaplan et al. 2024) doesn't wash away the consequences of segmenting of words into linguistically malformed chunks. Despite being trained on human-generated language, LLMs decompose words to idiosyncratic constituents, and they use those constituents to categorize words in idiosyncratic ways.

8. General Discussion

8.1 Summary

Large language models process text as a sequence of tokens, so they don't directly perceive meaningful constituents such as suffixes and semantic radicals. In some cases,

subword tokens align with meaningful constituents, but tokenization is determined by frequency, not semantics, so in many other cases, subword tokens distort or obscure meaningful constituents. The experiments reported in this study provide evidence that three state-of-the-art LLMs—GPT-4, GPT-4o, and Llama 3—draw inferences from sub-character tokens when representing the meanings of Chinese characters, including when those tokens are misleading. In Experiment 1, all three LLMs were more likely to decide that characters contained the same semantic radical if they began with the same token; in Experiment 2, they gave higher semantic similarity ratings to characters that began with the same token; and in Experiment 3, they were less likely to decide that a character was the odd one out of a group when it began with the same token as the other four group members. Shared initial tokens influenced explicit decisions, inflated perceived similarity, and guided categorization.

Ironically, the mistakes that GPT-4, GPT-4o, and Llama 3 make (e.g., incorrectly deciding that characters share a radical because they begin with the same token) demonstrate their intelligence. They recognize patterns in subword tokens, and they use those patterns to supplement other inferences, such as the assumption that Chinese characters with dissimilar meanings are less likely to share semantic radicals. The problem, then, is that byte-pair encoding sometimes supplies LLMs with unreliable information, and the trend in LLMs towards longer tokens will entrench this problem. In Experiment 4, GPT-4 and GPT-4o were more likely to correctly decide that a pair of characters contained the same semantic radical if the characters were segmented into three single-byte tokens than if they corresponded to a whole-character token. In Experiment 5, GPT-4 and GPT-4o were more likely to reasonably decide that a character with a different radical than the other four group members was the odd one out if it comprised three tokens rather than one token. This provides evidence that morpheme-like tokens obscure sub-morphemic constituents, which complicates research that emphasizes morphology in tokenization (e.g., Hofmann, Pierrehumbert, and Schütze 2021) and which may help explain why morphological tokenization doesn't always improve LLM performance (e.g., Gutiérrez, Sun, and Su 2023). Sub-morphemic constituents play an important role in human language processing (e.g., Bergen 2004; Ćwiek et al. 2022; Farmer, Christiansen, and Monaghan 2006), and LLMs use such constituents when they're available. Larger vocabularies of longer tokens will make those constituents available less often.

The imperfect correspondence between semantic radicals and initial tokens makes Chinese a convenient test case, but it might also make Chinese unrepresentative. Semantic radicals are analogous to category markers in other languages, so Experiment 6 conducted odd-one-out tasks in ten Indo-European languages and two Uralic languages. Consistent with the effects observed in Chinese, all three LLMs were more likely to decide that a target word with the same inflectional suffix as the four other group members was the odd one out if it ended with a different token than the other four, compared to if it ended with the same token. The effect was smaller in Llama 3 than in GPT-4 and GPT-4o, but Llama 3 was also less likely to reasonably decide that a target which ends with a different suffix is the odd one out. The effect of shared final tokens was significant in English for all three LLMs, which suggests that the problems introduced by linguistically malformed tokens will not be resolved by a surfeit of training data.

8.2 Implications

These findings challenge the claim that LLMs disregard misleading subword tokens when representing word meanings. Work on detokenization presents compelling

arguments that LLMs map word forms onto abstract, high-level representations and that words with negligible differences in form but substantial differences in tokenization share representations (e.g., *enc* + *odings* versus *EN* + *COD* + *INGS*, per GPT-4; Feucht et al. 2024; Kaplan et al. 2024). But as mentioned in the Introduction, detokenization doesn't address the question of where those representations come from. The experiments presented in this study provide evidence that characters with shared tokens come to share semantic features, reflected in inflated semantic similarity ratings and in a tendency to lump those characters together when categorizing. These experiments did not investigate whether subword tokens play an ongoing role in semantic processing, contra detokenization, or whether the inflated similarity of characters that share tokens is instead a vestige of inferences drawn during training, now incorporated into abstract representations. Future work should test these competing explanations.

The finding that subword tokens corrupt representations of word meanings is important because people interact with LLMs in natural language. LLMs are complex black boxes, so trying to control their behavior with computer code is unwieldy and ineffective (though see Bricken et al. 2023). Instead, getting an LLM to do what you want requires talking to it and assuming that it interprets words as intended (e.g., Sahoo et al. 2024), and systematically corrupted representations will impede people's ability to instruct and interpret LLMs. Developers should take this into account when considering tokenization methods for future models, especially given the results of Experiments 4 and 5, demonstrating how word- or morpheme-like tokens obscure important information. One possible solution is tokenizer-free models, which process text as a sequence of bytes (e.g., ByT5 and BLT; Pagnoni et al. 2024; Xue et al. 2022) or a sequence of Unicode points (e.g., CANINE; Clark et al. 2022). Note, however, that a Chinese character is a single Unicode point, so Unicode still obscures radicals implied by bytes, and even when breaking text into single bytes, spelling elides important information found in phonology (e.g., Nygaard, Herold, and Namy 2009). No tokenization method will overcome the fact that text strips language of essential qualities, and at least some aspects of meaning require that language be grounded in sensorimotor experience (e.g., Barsalou 1999; Harnad 1990), but Experiments 4 and 5 demonstrate that shorter tokens can mitigate the challenges of representing word meanings.

8.3 Limitations

The emphasis on semantic radicals and suffixes overlooks other ways in which word form reflects meaning. Most notably, sound symbolism influences word processing and the cultural evolution of languages. For example, English adjectives that refer to small objects often contain high front vowels (Winter and Perlman 2021), Austronesian words that refer to the nose often contain nasal consonants (Blunt 2003), and thousands of geographically and historically diverse languages share other such cues (Blasi et al. 2016). Sound symbolism tends to reflect sensory information (e.g., Dingemanse 2012; Winter 2019), so these patterns might help supplement the lack of sensorimotor experience available to computational models. Several recent studies have shown that various LLMs are sensitive to sound symbolism (Cai et al. 2024; Marklová et al. 2025; Trott 2024).

This study focused on characters in isolation, and the impact of misleading tokens will surely be reduced when words and characters are embedded in sentences and paragraphs. At the same time, from the perspective of an LLM, a linguistic context is nothing more than a collection of token embeddings, and contextualization refers to how those embeddings attend to one another and change one another. So, although richer contexts will dilute a subword constituent's contribution to the final representation of a word,

those richer contexts ultimately derive from representations of tokens. If subword tokens skew word meanings, then subword tokens will skew the representations of contexts derived from those meanings.

8.4 Conclusion

GPT-4, GPT-4o, and Llama 3 attend to semantic radicals when representing Chinese characters and attend to suffixes when representing words in twelve European languages. Because LLMs cannot directly perceive semantic radicals and other subword constituents, they draw inferences from tokens, and when those inferences are unreliable, GPT-4, GPT-4o, and Llama 3 misidentify radicals, miscategorize characters and words, and inflate semantic similarity. Longer tokens obscure subword constituents, which suggests that tokenization will increasingly corrupt representations in models with larger vocabularies comprising longer tokens, and the misleading impact of malformed tokens persists in English, which suggests that an abundance of training data will not resolve the problem, either. These findings matter because users and developers interact with large language models in natural language, yet the representations of word meanings in large language models are systematically skewed by tokenization.

References

Alegre, Maria and Peter Gordon. 1999. Frequency effects and the representational status of regular inflections. *Journal of Memory and Language*, 40(1):41–61. https://doi.org/10.1006/jmla.1998.2607

Amenta, Simona and Davide Crepaldi. 2012. Morphological processing as we know it: An analytical review of morphological effects in visual word identification. *Frontiers in Psychology*, 3:232. https://doi.org/10.3389/fpsyg.2012.00232, PubMed: 22807919

Arnett, Catherine, Pamela D. Rivière, Tyler A. Chang, and Sean Trott. 2024. Different tokenization schemes lead to comparable performance in Spanish number agreement. *arXiv preprint arXiv:2403.13754*. https://doi.org/10.18653/v1/2024.sigmorphon-1.4

Baayen, Rolf Harald. 2001. *Word Frequency Distributions*. Springer Dordrecht. https://doi.org/10.1007/978-94-010-0844-0

Baayen, Rolf Harald, Yu-Ying Chuang, Elnaz Shafaei-Bajestan, and James P. Blevins. 2019. The discriminative lexicon: A unified computational model for the lexicon and lexical processing in comprehension and production grounded not in (de) composition but in linear discriminative learning. *Complexity*, 2019(1):4895891. https://doi.org/10.1155/2019/4895891

Bai, Jinze, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Barsalou, L. W. 1999. Perceptual symbol systems. *Behavioral and Brain Sciences*. Cambridge University Press. https://doi.org/10.1017/S0140525X99002149, PubMed: 11301525

Batsuren, Khuyagbaatar, Gábor Bella, and Fausto Giunchiglia. 2021. Morphynet: A large multilingual database of derivational and inflectional morphology. In *Proceedings of the 18th Sigmorphon Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 39–48. https://doi.org/10.18653/v1/2021.sigmorphon-1.5

Batsuren, Khuyagbaatar, Ekaterina Vylomova, Verna Dankers, Tsetsuukhei Delgerbaatar, Omri Uzan, Yuval Pinter, and Gábor Bella. 2024. Evaluating subword tokenization: Alien subword composition and OOV generalization challenge. *arXiv preprint arXiv:2404.13292*.

Bergen, Benjamin K. 2004. The psychological reality of phonaesthemes. *Language*, 80(2):290–311. https://doi.org/10.1353/lan.2004.0056

Blasi, Damián E., Søren Wichmann, Harald Hammarström, Peter F. Stadler, and Morten H. Christiansen. 2016. Sound–meaning association biases evidenced across thousands of languages. *Proceedings of the National Academy of Sciences*, 113(39):10818–10823. https://doi.org/10.1073/pnas.1605782113, PubMed: 27621455

Blunt, Robert. 2003. The phonestheme ŋ-in Austronesian languages. *Oceanic Linguistics*, 42(1):187–212. https://doi.org/10.1353/ol.2003.0001

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146. https://doi.org/10.1162/tacl_a_00051

Bostrom, Kaj and Greg Durrett. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624. https://doi.org/10.18653/v1/2020.findings-emnlp.414

Bricken, Trenton, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, et al. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits* blog post. https://transformer-circuits.pub/2023/monosemantic-features

Bubeck, Sébastien, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with GPT-4. *arXiv preprint arXiv:2303.12712*.

Cai, Zhenguang, Xufeng Duan, David Haslett, Shuqi Wang, and Martin Pickering. 2024. Do large language models resemble humans in language use? In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 37–56. https://doi.org/10.18653/v1/2024.cmcl-1.4

Cassani, Giovanni, Yu-Ying Chuang, and Rolf Harald Baayen. 2020. On the semantics of nonwords and their lexical category. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 46(4):621. https://doi.org/10.1037/xlm0000747, PubMed: 31318232

Church, Kenneth Ward. 2020. Emerging trends: Subwords, seriously? *Natural Language Engineering*, 26(3):375–382. https://doi.org/10.1017/S1351324920000145

Clark, Jonathan H., Dan Garrette, Iulia Turc, and John Wieting. 2022. Canine: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91. https://doi.org/10.1162/tacl_a_00448

Contreras Kallens, Pablo and Morton H. Christiansen. 2020. Phonological cues to semantic class membership across hundreds of languages. *Proceedings of the 13th International Conference on the Evolution of Language*.

Cutler, Anne, John A. Hawkins, and Gary Gilligan. 1985. The suffixing preference: A processing explanation. *Linguistics*, 23(5):723–758. https://doi.org/10.1515/ling.1985.23.5.723

Ćwiek, Aleksandra, Susanne Fuchs, Christoph Draxler, Eva Liina Asu, Dan Dediu, Katri Hiovain, Shigeto Kawahara, Sofia Koutalidis, Manfred Krifka, Pärtel Lippus, et al. 2022. The bouba/kiki effect is robust across cultures and writing systems. *Philosophical Transactions of the Royal Society B*, 377(1841):20200390. https://doi.org/10.1098/rstb.2020.0390, PubMed: 34775818

Dingemanse, Mark. 2012. Advances in the cross-linguistic study of ideophones. *Language and Linguistics compass*, 6(10):654–672. https://doi.org/10.1002/lnc3.361

Dingemanse, Mark, Damián E. Blasi, Gary Lupyan, Morten H. Christiansen, and Padraic Monaghan. 2015. Arbitrariness, iconicity, and systematicity in language. *Trends in cognitive sciences*, 19(10):603–615. https://doi.org/10.1016/j.tics.2015.07.013, PubMed: 26412098

Dufour, Sophie. 2008. Phonological priming in auditory word recognition: When both controlled and automatic processes are responsible for the effects. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, 62(1):33. https://doi.org/10.1037/1196-1961.62.1.33, PubMed: 18473627

Edman, Lukas, Helmut Schmid, and Alexander Fraser. 2024. CUTE: Measuring LLMs' understanding of their tokens. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3017–3026. https://doi.org/10.18653/v1/2024.emnlp-main.177

Elhage, Nelson, Tristan Hume, Catherine Olsson, Neel Nanda, Tom Henighan, Scott Johnston, Sheer El Schowk, Nicholas Joseph, Nova DasSarma, Ben Mann, Danny Hernandez, Ndousse Kamal Askell, Amanda, Andy Jones, Dawn Drain, Anna Chen, Yuntao Bai, Deep Ganguli, Liane Lovitt, Zac Hatfield-Dodds, Tom

Conerly, Shauna Kravec, Stanislav Fort, Saurav Kadavath, Josh Jacobson, Eli Tran-Johnson, , Jared Kaplan, Jack Clark, Tom Brown, Sam McCandlish, Dario Amodei, and Christopher Olah. 2022. Softmax linear units. *Transformers Circuits Thread*. https://transformer-circuits.pub/2022/solu/index.html

Farmer, Thomas A., Morten H. Christiansen, and Padraic Monaghan. 2006. Phonological typicality influences on-line sentence comprehension. *Proceedings of the National Academy of Sciences*, 103(32):12203–12208. https://doi.org/10.1073/pnas.0602173103, PubMed: 16882728

Feldman, Laurie Beth and Witina W. T. Siok. 1999. Semantic radicals contribute to the visual identification of Chinese characters. *Journal of Memory and Language*, 40(4):559–576. https://doi.org/10.1006/jmla.1998.2629

Feucht, Sheridan, David Atkinson, Byron Wallace, and David Bau. 2024. Token erasure as a footprint of implicit vocabulary items in LLMs. *arXiv preprint arXiv:2406.20086*. https://doi.org/10.18653/v1/2024.emnlp-main.543

Gatti, Daniele, Marco Marelli, and Luca Rinaldi. 2023. Out-of-vocabulary but not meaningless: Evidence for semantic-priming effects in pseudoword processing. *Journal of Experimental Psychology: General*, 152(3):851. https://doi.org/10.1037/xge0001304, PubMed: 36174173

Goulden, Robin, Paul Nation, and John Read. 1990. How large can a receptive vocabulary be? *Applied Linguistics*, 11(4):341–363. https://doi.org/10.1093/applin/11.4.341

Grave, Édouard, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomáš Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Gurnee, Wes, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. 2023. Finding neurons in a haystack: Case studies with sparse probing. *Transactions on Machine Learning Research*. *arXiv: arXiv:2305.01610*.

Gutiérrez, Bernal Jiménez, Huan Sun, and Yu Su. 2023. Biomedical language models are robust to sub-optimal tokenization. In the *22nd Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks*, pages 350–362.

Harnad, Stevan. 1990. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1–3):335–346. https://doi.org/10.1016/0167-2789(90)90087-6

Haslett, David A. and Zhenguang G. Cai. 2023. Similar-sounding words flesh out fuzzy meanings. *Journal of Experimental Psychology: General*, 152(8):2359. https://doi.org/10.1037/xge0001409, PubMed: 37307335

Haslett, David A. and Zhenguang G. Cai. 2024. Systematic mappings of sound to meaning: A theoretical review. *Psychonomic Bulletin & Review*, 31(2):627–648. https://doi.org/10.3758/s13423-023-02395-y, PubMed: 37803232

Haslett, David A. and Zhenguang G. Cai. 2025. How much semantic information is available in large language model tokens? *Transactions of the Association for Computational Linguistics*. https://doi.org/10.1162/tacl_a_00747

Hendrix, Peter and Ching Chu Sun. 2021. A word or two about nonwords: Frequency, semantic neighborhood density, and orthography-to-semantics consistency effects for nonwords in the lexical decision task. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 47(1):157. https://doi.org/10.1037/xlm0000819, PubMed: 31999159

Hofmann, Valentin, Janet Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Derivational morphology improves BERT's interpretation of complex words. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3594–3608. https://doi.org/10.18653/v1/2021.acl-long.279

Itzhak, Itay and Omer Levy. 2022. Models in a spelling bee: Language models implicitly learn the character composition of tokens. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5061–5068. https://doi.org/10.18653/v1/2022.naacl-main.373

Jabbar, Haris. 2023. MorphPiece: Moving away from statistical language representation. *arXiv preprint arXiv:2307.07262*.

Kaplan, Guy, Matanel Oren, Yuval Reif, and Roy Schwartz. 2024. From tokens to words: On the inner lexicon of LLMs. *arXiv preprint arXiv:2410.05864*.

Kaushal, Ayush and Kyle Mahowald. 2022. What do tokens know about their characters and how do they know it? In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2487–2507. https://doi.org/10.18653/v1/2022.naacl-main.179

Kelly, Michael H. 1992. Using sound to solve syntactic problems: The role of phonology in grammatical category assignments. *Psychological Review*, 99(2):349. https://doi.org/10.1037//0033-295X.99.2.349, PubMed: 1594729

Kwon, Nahyun. 2017. Empirically observed iconicity levels of English phonaesthemes. *Public Journal of Semiotics*, 7(2):73–93. https://doi.org/10.37693/pjos.2016.7.16470

Lenci, Alessandro, Magnus Sahlgren, Patrick Jeuniaux, Amaru Cuba Gyllensten, and Martina Miliani. 2022. A comparative evaluation and analysis of three generations of distributional semantic models. *Language Resources and Evaluation*, 56(4):1269–1313. https://doi.org/10.1007/s10579-021-09575-z

Mager, Manuel, Arturo Oncevay, Elisabeth Maier, Katharina Kann, and Thang Vu. 2022. BPE vs. morphological segmentation: A case study on machine translation of four polysynthetic languages. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 961–971. https://doi.org/10.18653/v1/2022.findings-acl.78

Marelli, Marco, Simona Amenta, and Davide Crepaldi. 2015. Semantic transparency in free stems: The effect of orthography-semantics consistency on word recognition. *Quarterly Journal of Experimental Psychology*, 68(8):1571–1583. https://doi.org/10.1080/17470218.2014.959709, PubMed: 25269473

Marklová, Anna, Jiří Milička, Leonid Ryvkin, L'udmila Lacková Bennet, and Libuše Kormaníková. 2025. Iconicity in large language models. *arXiv preprint arXiv:2501.05643*.

Mielke, Sabrina J., Zaid Alyafeai, Elizabeth Salesky, Colin Raffel, Manan Dey, Matthias Gallé, Arun Raja, Chenglei Si, Wilson Y. Lee, Benoît Sagot, et al. 2021. Between words and characters: A brief history of open-vocabulary modeling and tokenization in NLP. *arXiv preprint arXiv:2112.10508*.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 26.

Nygaard, Lynne C., Debora S. Herold, and Laura L. Namy. 2009. The semantics of prosody: Acoustic and perceptual evidence of prosodic correlates to word meaning. *Cognitive Science*, 33(1):127–146. https://doi.org/10.1111/j.1551-6709.2008.01007.x, PubMed: 21585466

Pagnoni, Artidoro, Ram Pasunuru, Pedro Rodriguez, John Nguyen, Benjamin Muller, Margaret Li, Chunting Zhou, Lili Yu, Jason Weston, Luke Zettlemoyer, et al. 2024. Byte Latent Transformer: Patches scale better than tokens. *arXiv preprint arXiv:2412.09871*.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

Petrov, Aleksandar, Emanuele La Malfa, Philip Torr, and Adel Bibi. 2024. Language model tokenizers introduce unfairness between languages. *Advances in Neural Information Processing Systems*, 36.

R Core Team. 2024. R: A language and environment for statistical computing, R Foundation for Statistical Computing.

Sahoo, Pranab, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. 2024. A systematic survey of prompt engineering in large language models: Techniques and applications. *arXiv preprint arXiv:2402.07927*.

Schäfer, Anton, Thomas Hofmann, Imanol Schlag, and Tiago Pimentel. 2024. On the effect of (near) duplicate subwords in language modelling. *arXiv preprint arXiv:2404.06508*. https://doi.org/10.18653/v1/2024.findings-acl.571

Sennrich, Rico, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725. https://doi.org/10.18653/v1/P16-1162

Shi, Xinlei, Junjie Zhai, Xudong Yang, Zehua Xie, and Chao Liu. 2015. Radical embedding: Delving deeper to Chinese

radicals. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 594–598. https://doi.org/10.3115/v1/P15-2098

Si, Chenglei, Zhengyan Zhang, Yingfa Chen, Fanchao Qi, Xiaozhi Wang, Zhiyuan Liu, Yasheng Wang, Qun Liu, and Maosong Sun. 2023. Sub-character tokenization for Chinese pretrained language models. *Transactions of the Association for Computational Linguistics*, 11:469–487. https://doi.org/10.1162/tacl_a_00560

Speer, Robyn. 2022. rspeer/wordfreq: v3. 0. *Version v3. 0.2. Sept*.

Sun, Ching Chu, Peter Hendrix, Jianqiang Ma, and Rolf Harald Baayen. 2018. Chinese lexical database (cld) a large-scale lexical database for simplified mandarin Chinese. *Behavior Research Methods*, 50:2606–2629. https://doi.org/10.3758/s13428-018-1038-3, PubMed: 29934697

Sun, Zijun, Xiaoya Li, Xiaofei Sun, Yuxian Meng, Xiang Ao, Qing He, Fei Wu, and Jiwei Li. 2021. ChineseBERT: Chinese pretraining enhanced by glyph and pinyin information. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2065–2075. https://doi.org/10.18653/v1/2021.acl-long.161

Taft, Marcus and Xiaoping Zhu. 1997. Submorphemic processing in reading Chinese. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 23(3):761. https://doi.org/10.1037/0278-7393.23.3.761

Trott, Sean. 2024. Large language models and the wisdom of small crowds. *Open Mind*, 8:723–738. https://doi.org/10.1162/opmi_a_00144, PubMed: 38828431

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, Alex Hayes, Lionel Henry, Jim Hester, et al. 2019. Welcome to the tidyverse. *Journal of Open Source Software*, 4(43):1686. https://doi.org/10.21105/joss.01686

Winter, Bodo. 2019. *Sensory Linguistics*. John Benjamins Publishing Company. https://doi.org/10.1075/celcr.20.c1

Winter, Bodo and Marcus Perlman. 2021. Size sound symbolism in the English lexicon. *Glossa*, 6(1):79. https://doi.org/10.5334/gjgl.1646

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. https://doi.org/10.18653/v1/2020.emnlp-demos.6

Wu, Xiaofeng, Karl Stratos, and Wei Xu. 2024. The impact of visual information in Chinese characters: Evaluating large models' ability to recognize and utilize radicals. *arXiv preprint arXiv:2410.09013*.

Wu, Yonghui, Jiang Xu, Diogo Almeida, Carroll Wainwright, Pamula Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Xue, Linting, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306. https://doi.org/10.1162/tacl_a_00461